REMARKS

The claims have been further amended to define the scope of the invention more clearly.

It is noted that the basis for the amended wording regarding "accessing said data models to pull information from said data models about the structures of said source databases and said multi-dimensional data warehouse" is to found on page 6, second paragraph, of the specification as originally filed.

Claim Rejections 35 U.S.C. §103

Claims 1, 7, 13 and 16 were rejected under 35 U.S.C. §103 as unpatentable over Rosensteel et al. (U.S. Patent Number 6,167,405) in view of Coker (U.S. Patent Number 5,640,550).

It is respectfully submitted that these claims as amended herewith are clearly patentable over Rosensteel in view of Coker.

The Rosensteel reference

Rosensteel teaches a system in which data from a source database 18 is extracted, transferred, transformed and loaded into a target database 20 (column 4, lines 40-41). The target database 20 implements a data warehouse (column 4, lines 50-52). A Data Replication Manager (DRM) provides a visual interface allowing the system administrator to develop warehouse requests to move data from one database system to another (column 5, lines 5-8). A warehouse designer client component 12-4 captures the data models of the different databases (column 5, lines 18-21). A data model represents the objects that make up an application environment plus the relationships among those objects (column 5, lines 21-25).

The Coker reference

Coker describes a system which allows a COBOL program to access data from an SQL-oriented database. The COBOL source program (1) is first compiled to generate a COBOL object code file (3) and a data dictionary (4) (column 7, lines 35-41). The object code file is then executed by the runtime system (6). Whenever the runtime system encounters an input or output instruction such as a READ or WRITE, it passes the request to an interface (5) (column 7, lines 64-67). The interface automatically builds SQL instructions that the database management system can understand. As it builds these SQL instructions, it looks at the data dictionary, which associates the COBOL records with their fields (column 8, lines 1-7). For example, if the COBOL program specifies a READ, this is automatically translated into a database SQL query, and the data that is read from the database is automatically translated into a COBOL record (column 7, lines 6-16).

This provides a seamless interface between the COBOL program and the SQL database: as far as the COBOL program is aware, it is running COBOL I/O statements and getting COBOL data back in return, although in fact the data came from an SQL database (column 4, lines 6-13).

The Combination of Rosensteel and Coker

It is respectfully submitted that even if the teachings of Rosensteel and Coker were combined, this would still not have taught or suggested the present invention to a person skilled in the art.

Although Coker teaches compiling a COBOL source program (1) to generate an executable object code file (3), it should be noted that this compilation clearly does not involve accessing any data models to pull information from them about the structures of any databases. It is merely a conventional compilation.

The building of an SQL query by the interface (5) does involve looking at the data dictionary (4) which associates the COBOL records with their fields, but it is submitted that this data dictionary does not amount to an entity-relationship data model and does not provide information about the structures of databases.

It is agreed that when an SQL query is generated by the interface (5), it must be stored temporarily. However, there is no suggestion that the SQL queries generated by the interface (5) from a plurality of COBOL instructions are appended to a file so as to build up an executable file. On the contrary, it is clear that in Coker each SQL query must be executed immediately after it is generated, so that the database can be accessed and the requested I/O operation performed, before the next COBOL object code instruction can be executed.

It is noted that Coker mentions "directives" (column 12, line 64). However, it is clear that these directives are merely comments that guide the creation of the data dictionaries. There is no suggestion in Coker of processing these directives in the manner defined in the present claims.

Motivation to combine Rosensteel and Coker

It is respectfully submitted that a person skilled in the art would not have been motivated to combine the teachings of Rosensteel and Coker. Rosensteel (like the present invention) is concerned with the problem of initially creating and populating a data warehouse. Coker, on the other hand, is concerned with allowing a COBOL program to access and use data from an existing SQL database. Coker is not concerned with creating and populating a data warehouse or any other kind of database. Thus, Rosensteel and Coker are concerned with quite separate and distinct problems, and it is respectfully submitted that a person skilled in the art would clearly not even have considered combining their teachings in any way.

Combining the teaching of Rosensteel and Coker would necessarily involve writing a program in COBOL to create and populate a data warehouse, and then using the Coker's

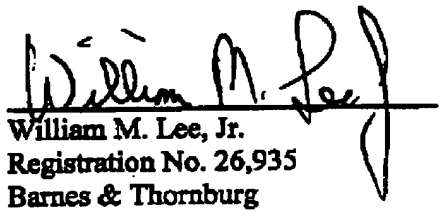
interface to convert each statement in the COBOL program into an SQL statement to allow it to access relational databases. But why would a person skilled in the art wish to write such a creation/population program in COBOL, rather than writing it in the first place in SQL? It is respectfully submitted that the skilled person could only have considered this with the benefit of hindsight, in the light of the present invention's teaching that it is advantageous to use high-level directives to write the creation/population program, and then to translate or convert these into the actual executable code, using information pulled from the data models.

Conclusion

In summary, it is respectfully submitted that this application is clearly in order for allowance and such action is respectfully solicited.

March 8, 2004

Respectfully submitted,



William M. Lee, Jr.
Registration No. 26,935
Barnes & Thornburg
P.O. Box 2786
Chicago, Illinois 60690-2786
(312) 214-4800
(312) 759-5646 (fax)